DECSAI

Departamento de Ciencias de la Computación e I.A.

Universidad de Granada

Perceptrones

Fernando Berzal, berzal@acm.org

Perceptrones

- Introducción
  - Redes neuronales artificiales
  - Modelos de redes
- Modelo de neurona artificial
  - Funciones de activación
- Perceptrones
  - La neurona de McCulloch y Pitts
  - El algoritmo de aprendizaje del perceptrón
  - Interpretación geométrica
  - Limitaciones
  - El perceptrón multiclase

Introducción

Redes Neuronales Artificiales [RNA]
Artificial Neural Networks [ANN]

Red de elementos o unidades de procesamiento simples (EP/UP en español o PE/PU en inglés), interconectados entre sí mediante conexiones sinápticas, en las que cada conexión tiene un peso (fuerza) que se ajusta a partir de la experiencia (datos).

Introducción

Redes Neuronales Artificiales [RNA]
Artificial Neural Networks [ANN]

- **Modelo de Neurona Artificial**:
  Modelo de cómputo simple. Cada neurona recibe estímulos de otras neuronas, los agrega y transmite una respuesta de acuerdo a su función de activación.

- **Modelo de Red Neuronal [Topología]**:
  Estructura de la red neuronal.
  Organización y número de neuronas y conexiones.

Introducción

Redes Neuronales Artificiales [RNA]
Artificial Neural Networks [ANN]

Las redes neuronales artificiales proporcionan un modelo de cómputo paralelo y distribuido capaz de aprender a partir de ejemplos (datos)

Los **algoritmos de aprendizaje** (asociados a modelos concretos de redes) permiten ir modificando los pesos de las conexiones sinápticas de forma que la red aprenda a partir de los ejemplos que se le presentan.

Introducción

Redes Neuronales Artificiales [RNA]
Artificial Neural Networks [ANN]

- No se programan, se entrenan.
- Necesitan disponer de ejemplos, en un número suficiente y una distribución representativa para ser capaces de generalizar correctamente.
- Requieren un proceso de validación para evaluar la "calidad" del aprendizaje conseguido.

Introducción

Redes Neuronales Artificiales [RNA]

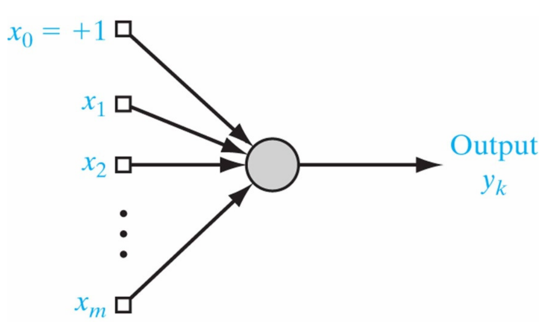
Artificial Neural Networks [ANN]

Veremos cómo…

- Entrenar redes neuronales artificiales (para distintos modelos de red).
- Preparar (preprocesar) los ejemplos necesarios para su entrenamiento.
- Evaluar la calidad del proceso de aprendizaje.

Introducción: Modelos de redes

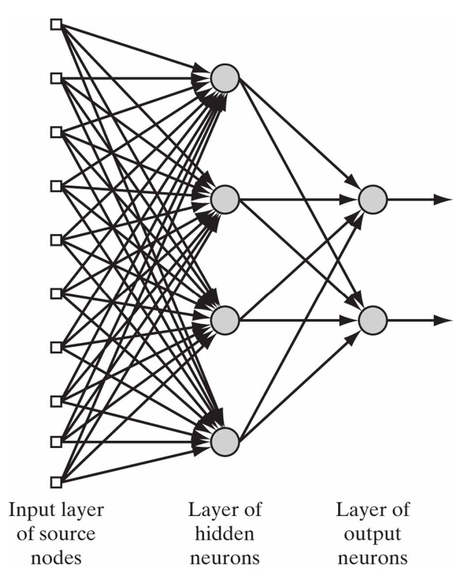
Perceptrón



[Haykin: "Neural Networks and Learning Machines", 3rd edition]

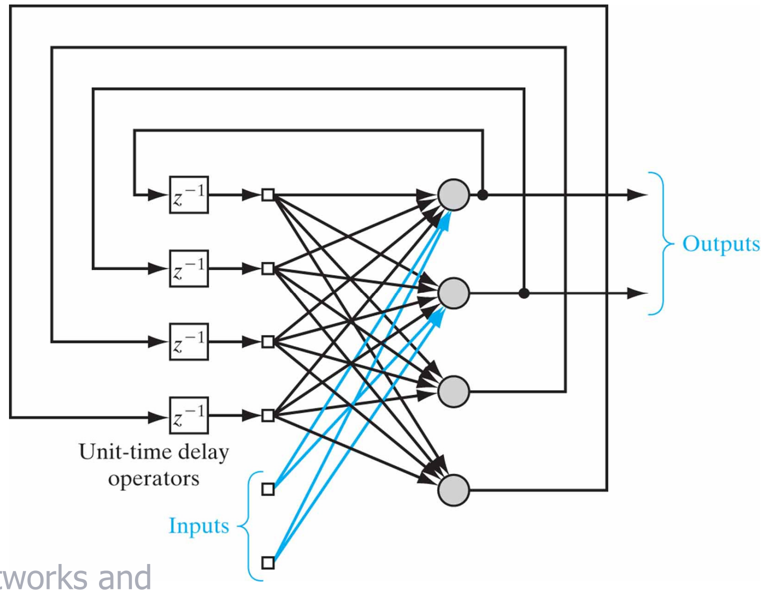
Redes feed-forward

(topología por capas)



Introducción: Modelos de redes

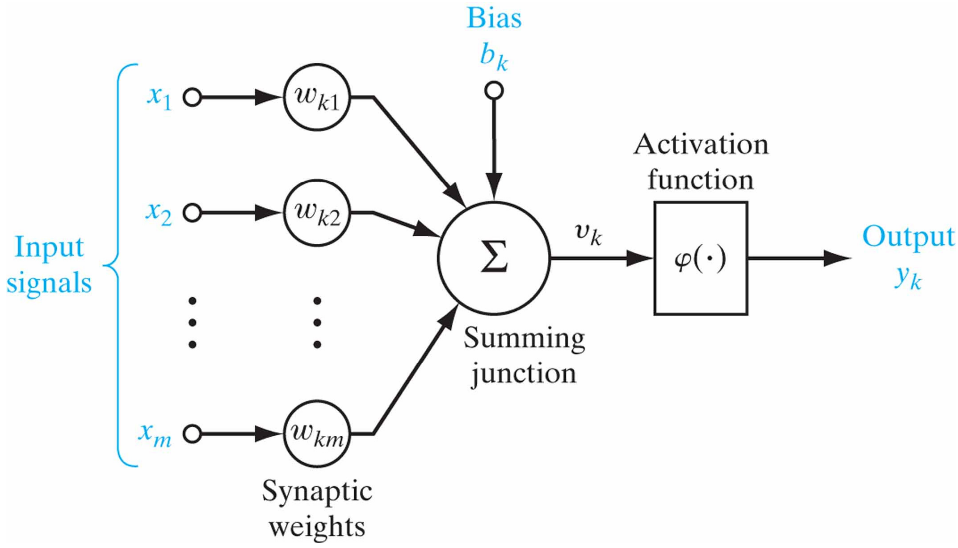
Redes recurrentes



[Haykin: "Neural Networks and Learning Machines", 3rd edition]

Modelo de neuronal artificial

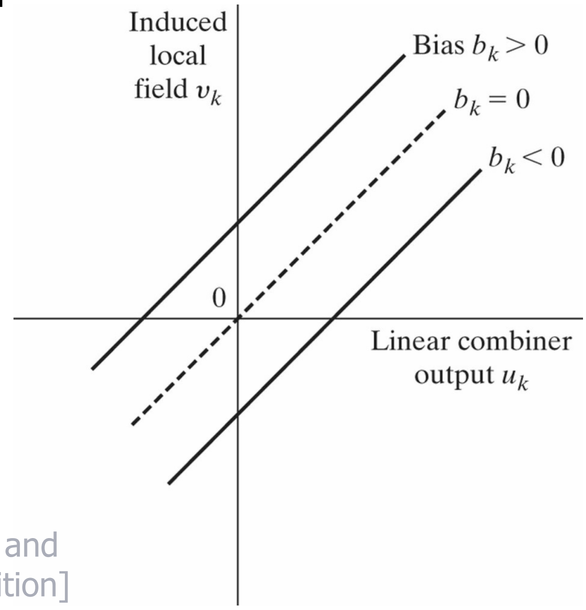
Modelo no lineal de una neurona artificial



[Haykin: "Neural Networks and Learning Machines", 3rd edition]

Modelo de neuronal artificial

Transformación afín producida por la presencia del **sesgo b_k [bias]**

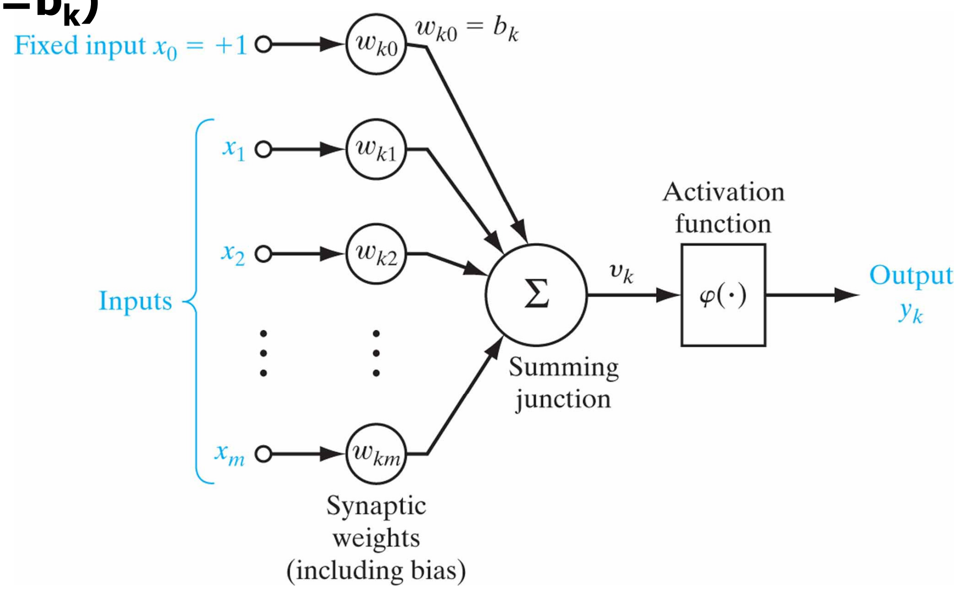


$v_k=b_k$ cuando $u_k=0$

[Haykin: "Neural Networks and Learning Machines", 3rd edition]

Modelo de neuronal artificial

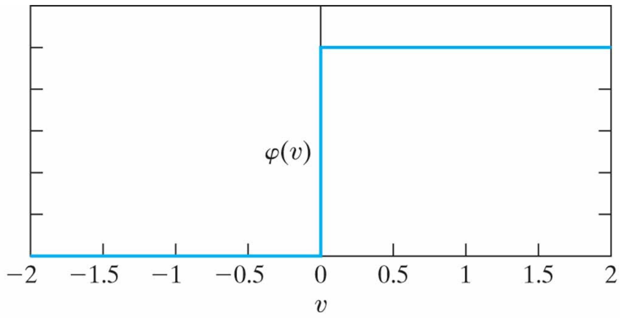
Modelo no lineal de una neurona artificial ($w_{k0}=b_k$)



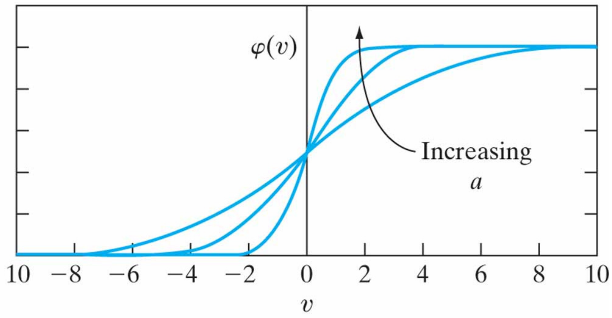
[Haykin: "Neural Networks and Learning Machines", 3rd edition]

Modelo de neuronal artificial

Funciones de activación φ(v)



Neuronas binarias



Neuronas sigmoidales

[Haykin: "Neural Networks and Learning Machines", 3rd edition]

Modelo de neurona artificial

Funciones de activación sigmoidales

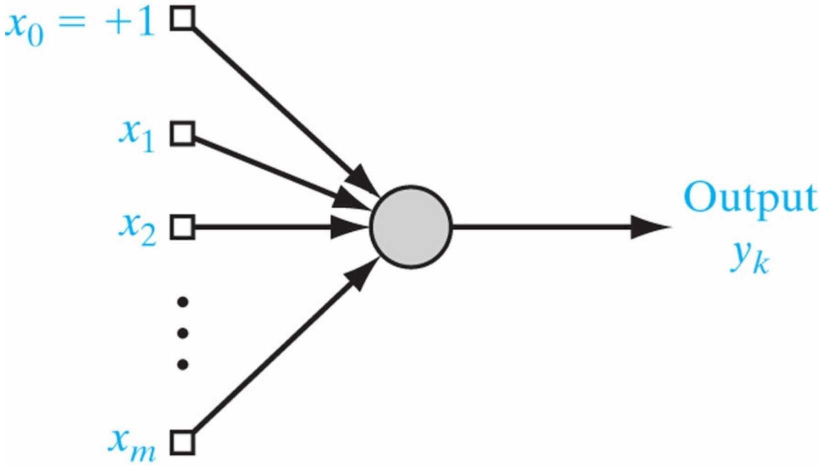
- Función logística [0,1]

$$\varphi(v) = \frac{1}{1 + e^{-av}}$$

- Tangente hiperbólica [-1,1]

$$\varphi(v) = \tanh v = \frac{e^{v} - e^{-v}}{e^{v} + e^{-v}}$$

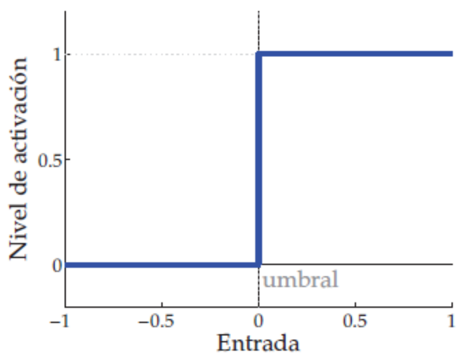
Perceptrones



Perceptrones

Modelo de neurona

Neuronas binarias con umbral
[McCulloch & Pitts, 1943]



$$z = \sum_i x_i w_i$$

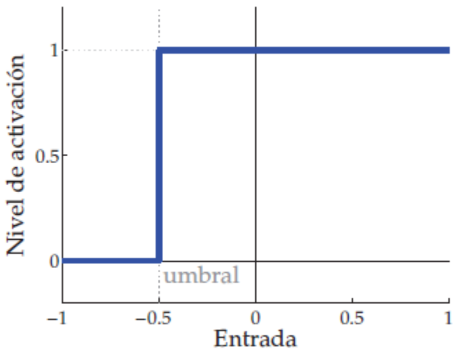
$$y = \begin{cases} 1 & si\ z \geq 0 \\ 0 & en\ otro\ caso \end{cases}$$

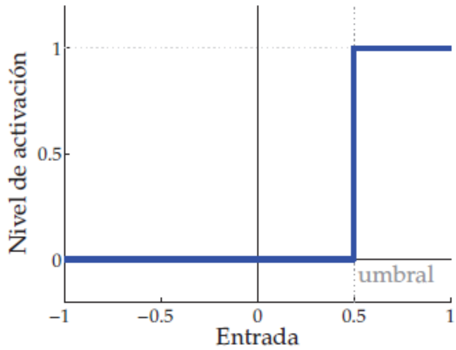
Asumiendo x_0=1 y w_0=b (umbral θ=-b)

Perceptrones

Modelo de neurona

Neuronas binarias con umbral



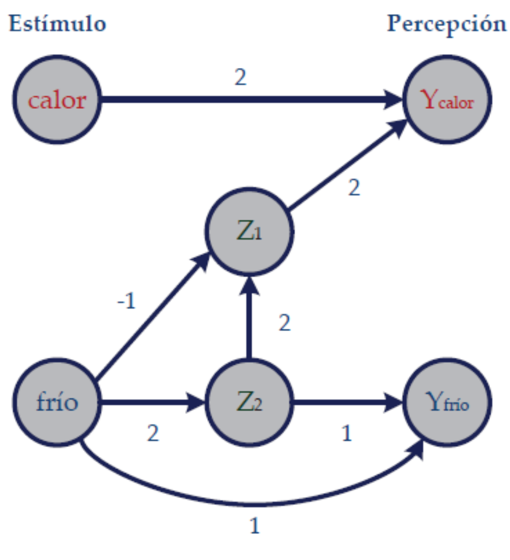


El efecto del umbral

Perceptrones

Modelo de neurona

Ejemplo: Percepción fisiológica del calor y del frío



Perceptrones

La primera generación de redes neuronales

- Popularizadas por Frank Rosenblatt en los años 60.
- Minsky y Papert analizaron lo que podían hacer y mostraron sus limitaciones en su libro de 1969.

Muchos pensaron que esas limitaciones se extendían a todos los modelos de redes neuronales, aunque no es así.

Su algoritmo de aprendizaje todavía se usa para tareas en las que los vectores de características contienen millones de elementos.

Perceptrones

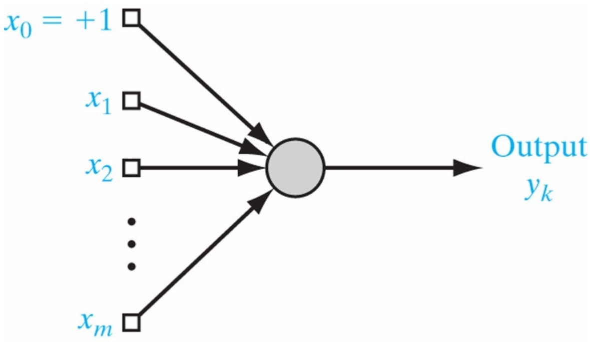
Reconocimiento de patrones

- Se convierten los datos de entrada en un vector de características x_j.
- Se aprenden los pesos asociados a cada una de esas características para obtener un valor escalar a partir de cada vector de entrada.
- Si este valor escalar se halla por encima de un umbral, se decide que el vector de entrada corresponde a un ejemplo de la clase objetivo (y_k=1).

Perceptrones

Algoritmo de aprendizaje

Un umbral (positivo) es equivalente a un sesgo/bias (negativo), por lo que podemos evitar tratar de forma separada el umbral añadiendo una entrada fija $x_0=+1$. De esta forma, aprendemos el umbral como si fuese un peso más.



Perceptrones

Algoritmo de aprendizaje

Se seleccionan ejemplos del conjunto de entrenamiento utilizando cualquier política que garantice que todos los ejemplos de entrenamiento se acabarán escogiendo:

- Si la salida es correcta, se dejan los pesos tal cual.
- Si la unidad de salida incorrectamente da un cero, se añade el vector de entrada al vector de pesos.
- Si la unidad de salida incorrectamente da un uno, se resta el vector de entrada del vector de pesos.

Perceptrones

TABLE 1.1 Summary of the Perceptron Convergence Algorithm

Variables and Parameters:

$\mathbf{x}(n) = (m+1)$-by-1 input vector
$= [+1, x_1(n), x_2(n), \ldots, x_m(n)]^T$
$\mathbf{w}(n) = (m+1)$-by-1 weight vector
$= [b, w_1(n), w_2(n), \ldots, w_m(n)]^T$
b = bias
$y(n)$ = actual response (quantized)
$d(n)$ = desired response
η = learning-rate parameter, a positive constant less than unity

1. *Initialization.* Set $\mathbf{w}(0) = \mathbf{0}$. Then perform the following computations for time-step $n = 1, 2, \ldots$.
2. *Activation.* At time-step n, activate the perceptron by applying continuous-valued input vector $\mathbf{x}(n)$ and desired response $d(n)$.
3. *Computation of Actual Response.* Compute the actual response of the perceptron as
$$y(n) = \operatorname{sgn}[\mathbf{w}^T(n)\mathbf{x}(n)]$$
where $\operatorname{sgn}(\cdot)$ is the signum function.
4. *Adaptation of Weight Vector.* Update the weight vector of the perceptron to obtain
$$\mathbf{w}(n+1) = \mathbf{w}(n) + \eta[d(n) - y(n)]\mathbf{x}(n)$$
where
$$d(n) = \begin{cases} +1 & \text{if } \mathbf{x}(n) \text{ belongs to class } \mathscr{C}_1 \\ -1 & \text{if } \mathbf{x}(n) \text{ belongs to class } \mathscr{C}_2 \end{cases}$$
5. *Continuation.* Increment time step n by one and go back to step 2.

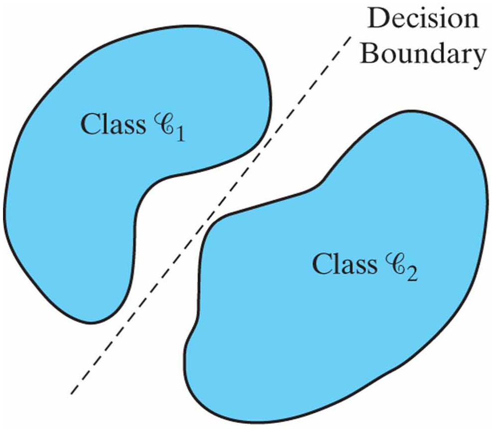
[Haykin: "Neural Networks and Learning Machines", 3rd edition]

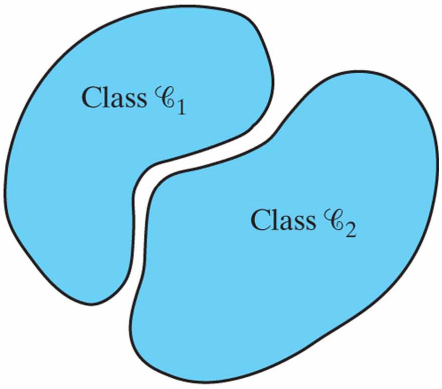
Perceptrones

Algoritmo de aprendizaje

- El algoritmo de aprendizaje del perceptrón garantiza encontrar un conjunto de pesos que proporcione la respuesta correcta si tal conjunto existe.
- El perceptrón es un modelo de clasificación lineal, por lo cual será capaz de clasificar correctamente los ejemplos de entrada siempre que las clases sean linealmente separables.

Perceptrones



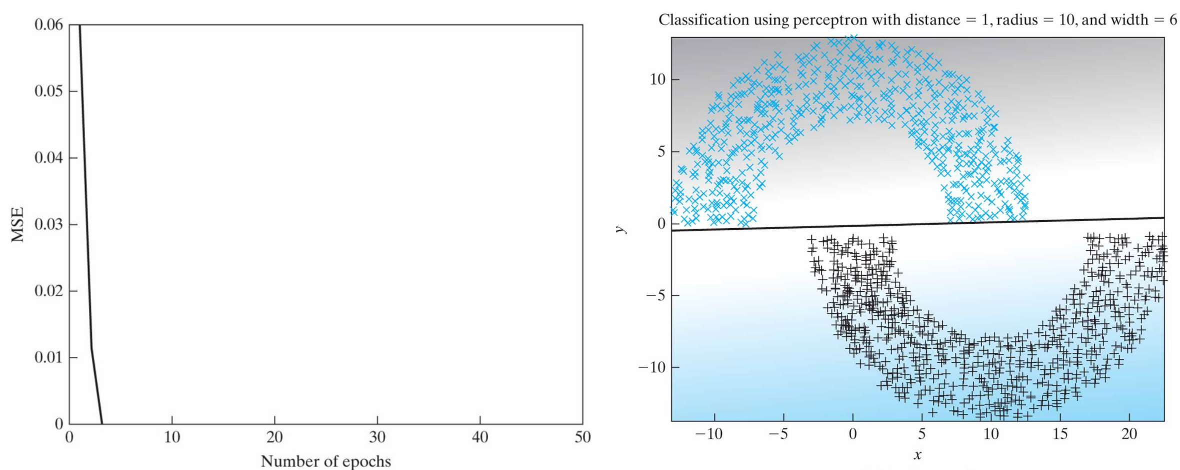


Clases
linealmente separables

Clases
linealmente no separables

[Haykin: "Neural Networks and Learning Machines", 3rd edition]

Perceptrones

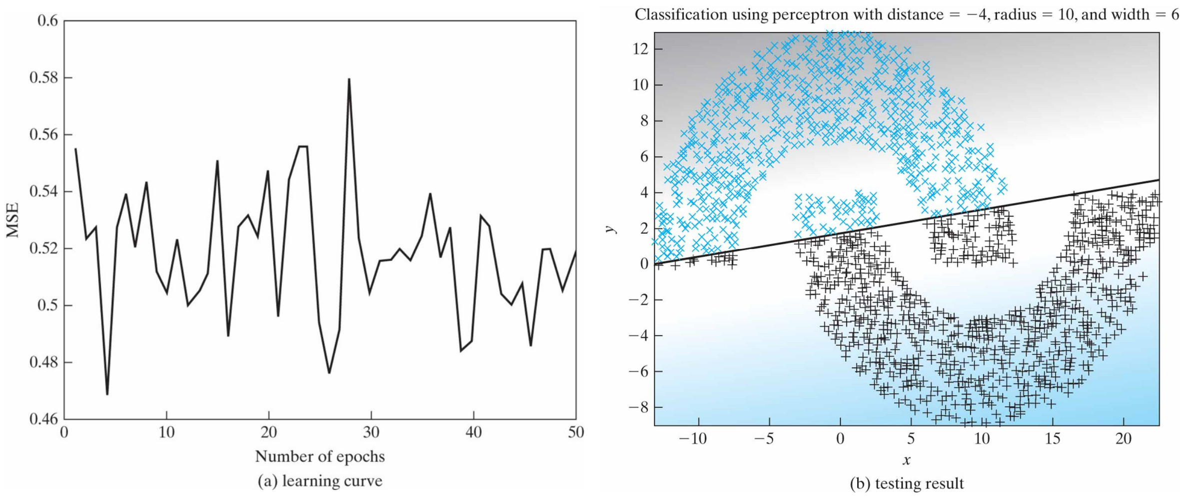


(a) learning curve (b) testing result

Clases linealmente separables

[Haykin: "Neural Networks and Learning Machines", 3rd edition]

Perceptrones

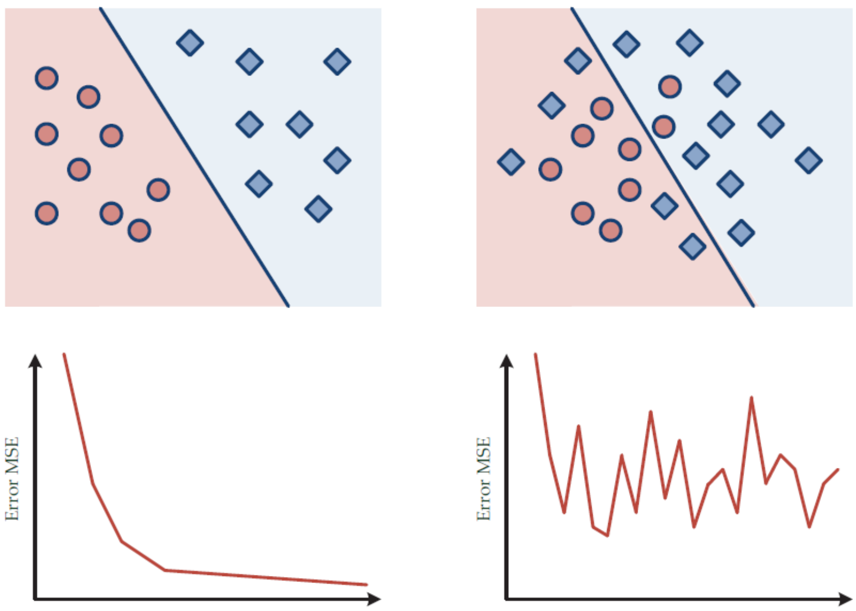


Clases linealmente no separables

[Haykin: “Neural Networks and Learning Machines”, 3rd edition]

Perceptrones

Clases linealmente separables
vs. Clases no linealmente separables



Perceptrones

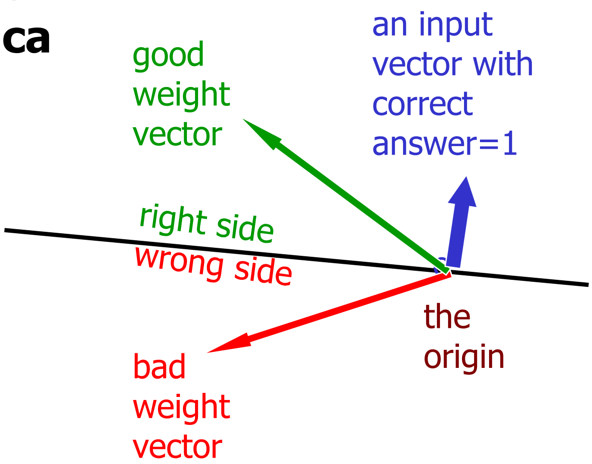
Algoritmo de aprendizaje:
Interpretación geométrica

Espacio de pesos:

- Una dimensión por cada peso.
- Cada punto del espacio representa
  un valor particular para el conjunto de pesos.
- Cada caso de entrenamiento corresponde
  a un hiperplano que pasa por el origen
  (tras eliminar el umbral e incluirlo como un peso más)

Perceptrones

Algoritmo de aprendizaje:
Interpretación geométrica

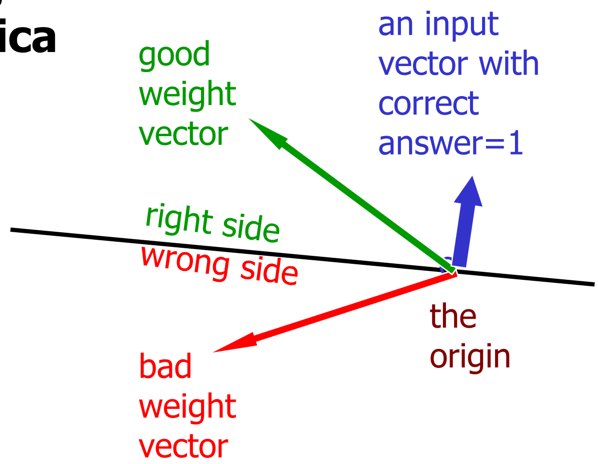


Cada caso define un hiperplano:

- Hiperplano perpendicular al vector de entrada.
- Los pesos deben quedar a un lado del hiperplano.

Perceptrones

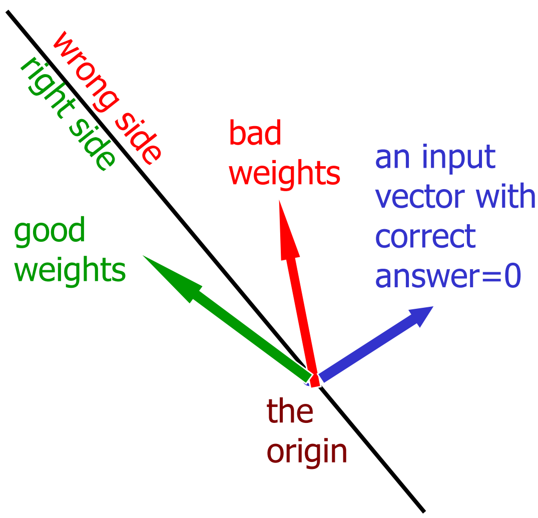
Algoritmo de aprendizaje: Interpretación geométrica



Si el producto escalar del vector de entrada con el vector de pesos es negativo, la salida será la equivocada.

Perceptrones

Algoritmo de aprendizaje: Interpretación geométrica

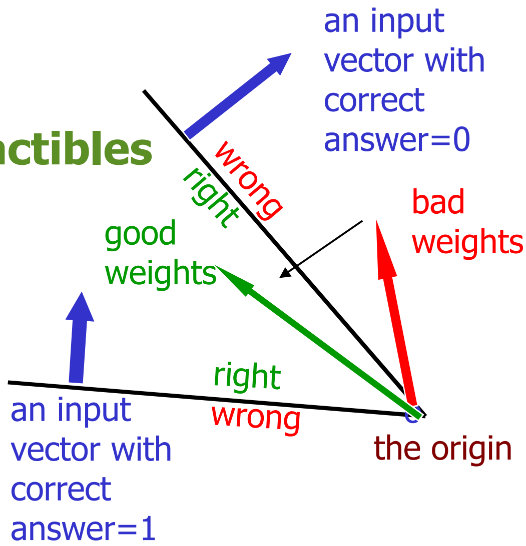


Si el producto escalar del vector de entrada con el vector de pesos es negativo, la salida será la equivocada.

Perceptrones

Algoritmo de aprendizaje: Interpretación geométrica

El hipercono de soluciones factibles

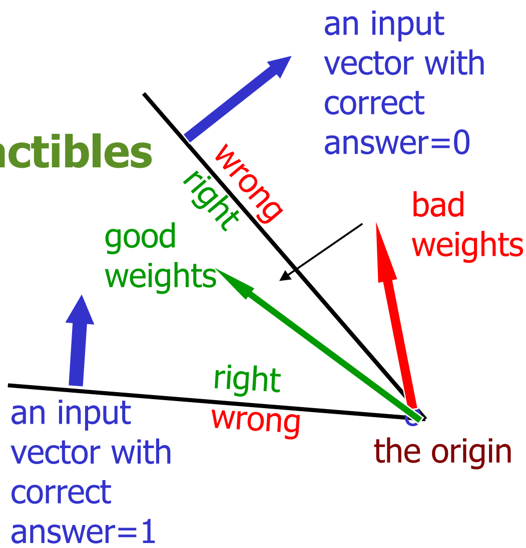


Para que el aprendizaje sea correcto, debemos encontrar un punto que esté en el lado correcto de todos los hiperplanos (que puede que no exista!!!).

Perceptrones

Algoritmo de aprendizaje: Interpretación geométrica

El hipercono de soluciones factibles

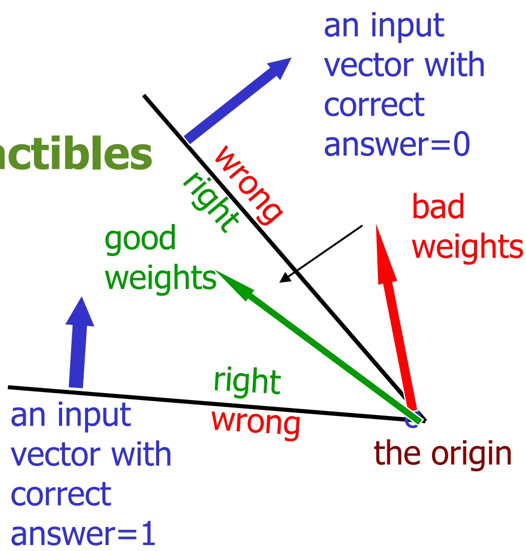


Si existe un conjunto de pesos que proporcionen la respuesta adecuada para todos los casos,
estará en un hipercono con su ápice en el origen.

Perceptrones

Algoritmo de aprendizaje:
Interpretación geométrica

El hipercono de soluciones factibles



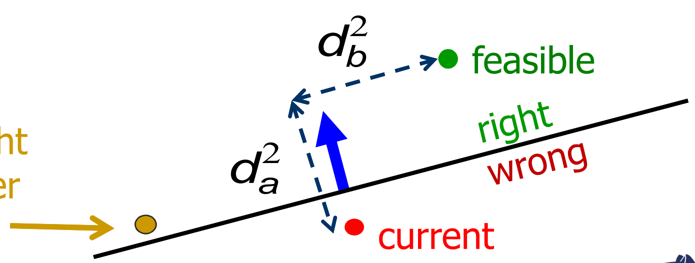
Al definir el hipercono de soluciones factibles una región convexa, la media de dos buenos vectores de pesos será también un buen vector de pesos…

Perceptrones

Algoritmo de aprendizaje:
Corrección del algoritmo

Hipótesis errónea: Cada vez que el perceptrón comete un error, el algoritmo de aprendizaje acerca el vector de pesos actual hacia todas las soluciones factibles.

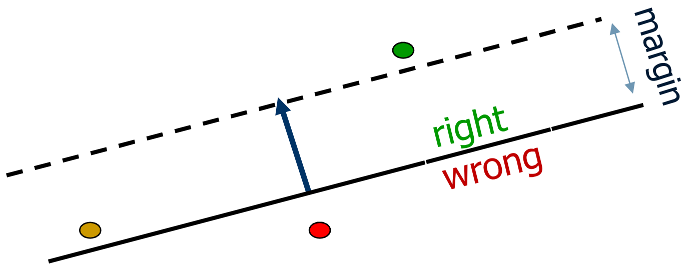
Problem case: The weight vector may not get closer to this feasible vector!



Perceptrones

Algoritmo de aprendizaje:
Corrección del algoritmo

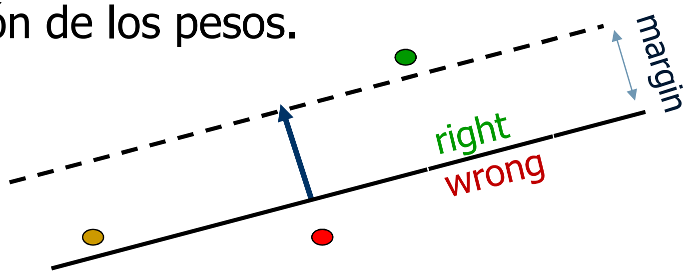
Consideramos los vectores de pesos “generosamente factibles” que quedan, dentro de la región factible, con un margen al menos tan grande como la longitud del vector de entrada.



Perceptrones

Algoritmo de aprendizaje:
Corrección del algoritmo

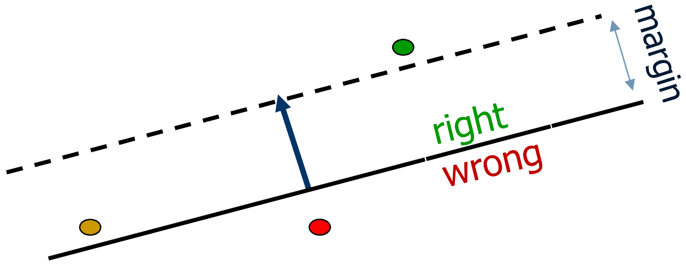
Cada vez que el perceptrón se equivoca,
el cuadrado de la distancia a todos esos vectores de pesos “generosamente factibles” siempre se decrementa en, al menos, el cuadrado de la longitud del vector de actualización de los pesos.



Perceptrones

Algoritmo de aprendizaje:
Corrección del algoritmo

Por tanto, tras un número finito de errores,
el vector de pesos deberá estar en la región factible,
si ésta existe.



Perceptrones

Limitaciones

- Si se pueden elegir todas las características que deseemos, se puede hacer cualquier cosa.

  p.ej. Con una entrada para cada posible vector (2^n),
  se puede discriminar cualquier función booleana,
  si bien el perceptrón no generalizará bien.

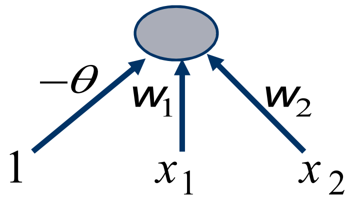
- Si las entradas vienen determinadas, existen severas limitaciones sobre lo que un perceptrón puede aprender (p.ej. XOR y EQ).

Perceptrones

Limitaciones

Un perceptrón no puede decidir si dos bits son iguales:

x_1	x_2	Y
0	0	1
0	1	0
1	0	0
1	1	1



4 casos definen 4 desigualdades imposibles de satisfacer:

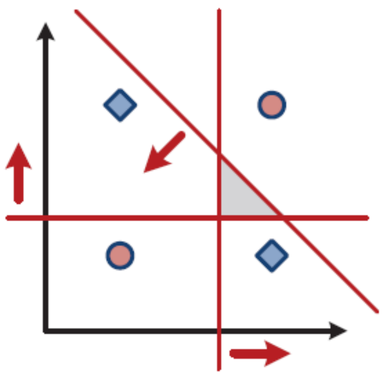
$$w_1 + w_2 \geq \theta, \quad 0 \geq \theta$$
$$w_1 < \theta, \quad w_2 < \theta$$

Perceptrones

Limitaciones

Un perceptrón no puede decidir si dos bits son iguales:

La función XOR define
un sistema de inecuaciones sin solución:



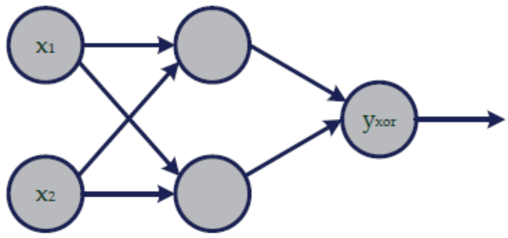
Perceptrones

Limitaciones

Un perceptrón no puede decidir si dos bits son iguales…

… pero sí podemos hacerlo con 2 capas de perceptrones:

Neurona	Entradas		Umbral θ
y_{00}	x_1	x_2	0.5
y_{11}	x_1	x_2	1.5
y_{xor}	$0.6y_{00}$	$-0.2y_{11}$	0.5



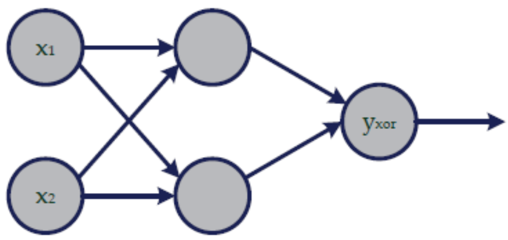
Perceptrones

Limitaciones

Un perceptrón no puede decidir si dos bits son iguales…

… y, además, de varias formas diferentes:

Neurona	Entradas		Umbral θ
y_{10}	$2x_1$	$-1x_2$	2
y_{01}	$-1x_1$	$2x_2$	2
y_{xor}	$2y_{10}$	$2y_{01}$	2

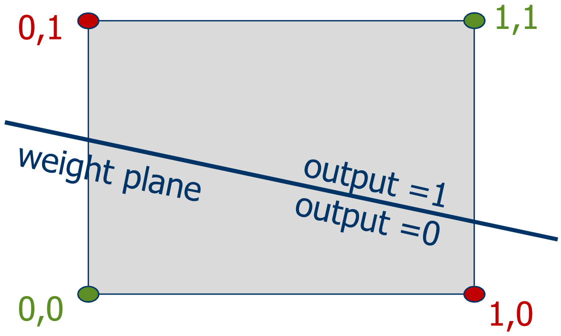


Perceptrones

Limitaciones:

Interpretación geométrica

Casos positivos y negativos NO pueden separarse por un plano



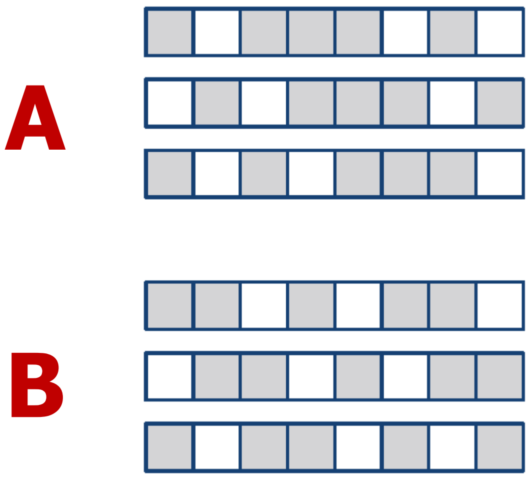
Espacio de datos:

- Una dimensión por cada característica.
- Un vector de entrada es un punto.
- Un vector de pesos define un hiperplano.
- El hiperplano es perpendicular al vector de pesos y está a una distancia del origen dada por el umbral θ.

Perceptrones

Limitaciones

Diferenciar entre diferentes patrones que tengan el mismo número de píxeles tampoco se puede si se admiten traslaciones:



Perceptrones

Limitaciones

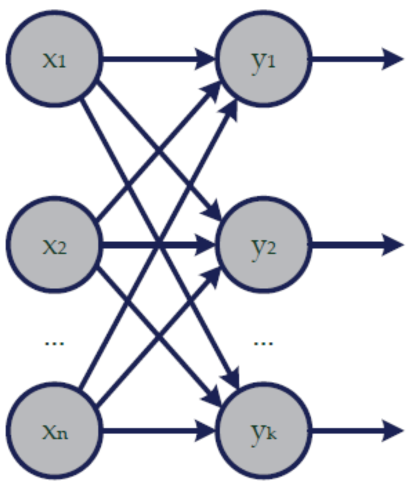
- El **teorema de invarianza de grupos** de Minsky y Papert establece que la parte del perceptrón que aprende no es capaz de reconocer patrones si las transformaciones a las que pueden estar sometidos dichos patrones forman un grupo.

- La parte interesante del reconocimiento de patrones debe resolverse manualmente (añadiendo nuevas características), pero no puede aprenderse usando un perceptrón…

Perceptrones

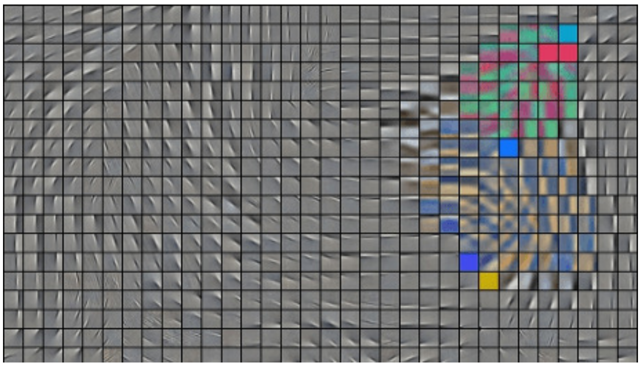
El perceptrón multiclase

Un perceptrón para cada clase del problema…



Referencias

Neural Networks for Machine Learning
by Geoffrey Hinton
(University of Toronto & Google)
https://www.coursera.org/course/neuralnets



Bibliografía

Lecturas recomendadas

- Fernando Berzal:
  Redes Neuronales & Deep Learning

  CAPÍTULO 7
  Perceptrones